Log de Transaçào por **cibellecastro -** <http://cibellecastro.wordpress.com/>

O que é Log de Transações?

É o responsável em registrar todas as transações e modificações feitas no banco de dados a cada transação. Segundo a definição encontrada no site da MSDN “O log de transações é um componente crítico do banco de dados e, se houver uma falha do sistema, será necessário que o log de transações retorne seu banco de dados a um estado consistente. O log de transações nunca deve ser excluído ou movido a menos que você compreenda plenamente as consequências disso.”

Cada alteração feita em um banco de dados tem uma entrada inserida no log de transação. Cada linha recebe um número exclusivo internamente, chamado LSN(Log Sequence Number). O LSN é um valor inteiro que começa em 0(zero) quando o banco de dados é criado e é incrementado até o infinito. Um LSN nunca é reutilizado e é sempre incrementado. podemos dizer que basicamente ele fornece um número em sequência para cada alteração feita em um banco de dados.

Mas e como é o conteúdo do log de transação? O conteúdo é dividido em duas partes. Uma ATIVA e uma INATIVA. Simples não é? A diferença entre as duas é bem lógica. A parte INATIVA contém todas as alterações “commitadas” no banco de dados. Enquanto a parte ATIVA do log contém todas as alterações ainda não “commitadas”.

Suponhamos que você resolve fazer um backup do log de transações, como o SQL Server vai agir? Quando o backup do log é iniciado o SQL Server inicia no menor LSN do log de transação e começa a gravar cada registro do log de transação  sucessivo no backup. Assim que o SQL Server atinge o primeiro  LSN que ainda não foi “commitado”(ou seja, a transação mais antiga aberta), o backup do log de transações termina.

A parte do log de transação que teve o backup feito é removida e libera espaço para ser reutilizado.

Um Log de Transações é formado de um ou mais arquivos associados a um banco de dados que monitoram toda modificação feita nos dados ou objetos dentro do “database”. O Log não ultrapassa os limites dos banco de dados . Por tanto uma transação empresarial executada em vários bancos de dados é fisicamente implementada como uma transação separada dentro de cada banco de dados afetado. O Log de Transações também é exigido para armazenar informações suficientes para permitir que o SQL Server recupere um banco de dados quando a instância for reiniciada.

Acima eu havia falado sobre LSN(Log Sequence Number), ele é a principal informação dentro do Log. Cada operação que afete o banco de dados incrementa o LSN. Cada unidade de armazenamento dentro do “database” monitora o LSN da última modificação feita na estrutura de armazenamento. Em nível de banco de dados, o LSN da última alteração feita no banco é armazenado no cabeçalho do master data file(arquivo de dados mestre). A nível de arquivo de dados(data files), o LSN da última alteração feita em uma página dentro do arquivo é armazenado no cabeçalho do arquivo de dados. Cada página de dados dentro de um banco de dados também grava o LSN da última alteração da página de dados.  
Todas as alterações de dados ocorrem dentro do buffer na memória. Quando é feito uma alteração, o buffer correspondente é modificado e o registro é adicionado no log de transação. Uma página modificada no buffer pool é referenciada como uma dirty page(página suja). Cada dirty page monitora o LSN no Log de Transação correspondente à alteração que modificou a pagina no buffer pool. Quando o SQL Server executa um checkpoint, todas as páginas sujas do buffer pool são gravadas nos arquivos de dados.  
Durante o processo de checkpoint, o SQL Server compara o LSN da página suja no buffer pool com o LSN da página de dados no disco. Se o LSN da página de dados no disco é igual ou menor do que o LSN da página suja no buffer pool, assim como igual ou menor do que o LSN do arquivo de dados, a página do disco é sobrescrita pela página do buffer pool. Se o LSN da dirty page é maior do que o da página de disco ou do arquivo de dados que contém a página, a página do buffer pool é sobrescrita pela página do disco.





Quando o processo de checkpoint termina de gravar a páginas sujas nos arquivos de dados , o maior LSN gravado em cada arquivo é gravado no cabeçalho do mesmo. Além disso o maior LSN gravado no processo de checkpoint inteiro é gravado no cabeçalho do arquivo de dados mestre(master data ﬁle).

O SQL Server garante que o LSN de cada página dentro de uma arquivo seja igual ou menor do que o LSN do arquivo. e que o LSN de cada arquivo dentro do banco de dados seja igual ou menor do que o LSN dentro do banco de dados. A última etapa do processo é apagar, em cada dirty page afetada pelo chekpoint, o flag que identifica que a página foi alterada.

Quando o SQL Server é iniciado, cada banco de dados passa por um processo chamado de RESTART RECOVERY. O restart recovery ocorre em duas fases. Essas fases são chamadas de ***UNDO*** e ***REDO.***

Durante a fase do REDO, todas as transações “commitadas” do LOG de Transações são descarregadas no disco. O REDO usa a mesma lógica do processo de checkpoint. Se o LSN armazenado na page(página) é menor ou igual ao LSN  do log record(resgistro de log)que esta sendo gravado na página, a alteração é gravada. Caso contrário, é pulada, sendo considerada como já existente no disco. Quando a fase do REDO termina, a fase UNDO começa.

A fase do UNDO percorre o LOG de Transação e invalida qualquer transação no log que ainda esteja aberta, garantindo que uma transação uncommitted(não efetivada) não possa ser gravada no disco. Quando a fase do UNDO termina, o banco de dados passa por um processo referenciado como rolling forward.

Quando um banco de dados esta no processo de rolling forward , o SQL Server lê o último LSN gravado no LOG de Transação, incrementa o LSN e grava o novo LSN no cabeçalho de todos os data file(arquivo de dados) garantindo as transações  mais antigas do que o ponto  roll-forward não possam ser gravadas nos data files.

Todo backup criado armazena o LSN mínimo e o máximo do banco de dados o qual corresponde o backup taken.

Como um backup FULL contém a porção do log de transação que foi gerado enquanto o backup está em execução, um backup FULL é consistente com o momento da sua conclusão e armazena apenas o último LSN utilizado dentro do backup. Os backups diferenciais e de log de transação gravam o LSN do banco de dados no início da operação de backup, bem como o LSN, no final da operação de backup.

Como o LSN esta sempre avançando (moving forward), o SQL Server só precisa comparar o LSN atual com o LSN(ou LSNs) gravado para o backup e assim determinar se o backup pode ser aplicado no banco de dados. Se o backup possui o próximo LSN da sequência, então pode ser restaurado. Se o backup não contém o próximo LSN da sequência, um erro é gerado e o processo  de restauração termina sem aplicar quaisquer alteração.

Com o passar dos anos, muitas pessoas têm sido informadas incorretamente de que um backup diferencial e de log de transação podem ser restaurados em um banco de dados que é recuperado porque, no final do processo de restauração, o LSN é avançado (rolled forward) e não é mais compatível com nenhum dos backup de log ou diferencial.

Nesse caso o SQL Server não rejeita o backup diferencial e de log de transação devido ao LSN. Lembrando que um backup diferencial e de log são específicos para um backup full ou de filegroup.

Um banco de dados que é “recovered” pode ter transações executadas, o que deixa o banco de dados incompatível com o backup diferencial ou o backup do log de transação. Isso se deve porque as transações não podem ser executadas em um banco de dados ou filegroup que está em um estado “recovering”, SQL Server só tem de verificar se ambos estão em estado “recovering” para prosseguir com a verificação secundária de compatibilidade do LSN.